



UNITED STATES PATENT AND TRADEMARK OFFICE

JS

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/001,478	11/01/2001	Craig Nemecek	CYPR-CD01213M	6435

7590 09/05/2006

WAGNER, MURABITO & HAO LLP
Third Floor
Two North Market Street
San Jose, CA 95113

EXAMINER

PROCTOR, JASON SCOTT

ART UNIT	PAPER NUMBER
----------	--------------

2123

DATE MAILED: 09/05/2006

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	10/001,478	NEMECEK ET AL.	
	Examiner	Art Unit	
	Jason Proctor	2123	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

- 1) ☒ Responsive to communication(s) filed on 19 June 2006.
- 2a) ☒ This action is **FINAL**. 2b) ☐ This action is non-final.
- 3) ☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

- 4) ☒ Claim(s) 1,2,4-10,12 and 14-21 is/are pending in the application.
- 4a) Of the above claim(s) _____ is/are withdrawn from consideration.
- 5) ☐ Claim(s) _____ is/are allowed.
- 6) ☒ Claim(s) 1,2,4-10,12 and 14-21 is/are rejected.
- 7) ☐ Claim(s) _____ is/are objected to.
- 8) ☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

- 9) ☐ The specification is objected to by the Examiner.
- 10) ☒ The drawing(s) filed on 18 January 2001 is/are: a) ☒ accepted or b) ☐ objected to by the Examiner.
Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).
Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).
- 11) ☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

- 12) ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
- a) ☐ All b) ☐ Some * c) ☐ None of:
1. ☐ Certified copies of the priority documents have been received.
2. ☐ Certified copies of the priority documents have been received in Application No. _____.
3. ☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

- | | |
|--|---|
| 1) <input type="checkbox"/> Notice of References Cited (PTO-892) | 4) <input type="checkbox"/> Interview Summary (PTO-413) |
| 2) <input type="checkbox"/> Notice of Draftsperson's Patent Drawing Review (PTO-948) | Paper No(s)/Mail Date. _____ |
| 3) <input type="checkbox"/> Information Disclosure Statement(s) (PTO/SB/08) | 5) <input type="checkbox"/> Notice of Informal Patent Application |
| Paper No(s)/Mail Date _____ | 6) <input type="checkbox"/> Other: _____ |

DETAILED ACTION

Claims 1-10 and 12-21 were rejected in the Office Action of 15 March 2006. Applicants' response on 19 June 2006 has cancelled claims 3, 11, and 13. Claims 1-2, 4-10, 12, and 14-21 are pending in this application.

Claims 1-2, 4-10, 12, and 14-21 are rejected.

Priority

1. This Application contains a claim for the benefit of priority to U.S. Provisional Application No. 60/243,708 filed 26 October 2000. The provisional application has been reviewed and priority is denied, because the provisional application does not appear to enable the claimed invention as required under 35 U.S.C. Section 112, first paragraph. See 35 U.S.C. § 119(e)(1).

For example, the provisional application contains a set of 'powerpoint-style' drawings and datasheets describing desired features for a microcontroller or a 'system-on-chip,' but this material does not appear to contain either the text description or the drawings found in the Application. In particular, no part of the provisional application appears to disclose the method steps shown in the Application at Fig. 7.

Claim Objections

2. The previous objections to claims 3 and 14 have been withdrawn in response to the cancellation of those claims.

Claim Rejections - 35 USC § 112

3. The previous rejections of claims 3 and 13 under 35 U.S.C. § 112, second paragraph, are withdrawn in response to the cancellation of those claims.

Claim Rejections - 35 USC § 102

The following is a quotation of the appropriate paragraphs of 35 U.S.C. § 102 that form the basis for the rejections under this section made in this Office action:

A person shall be entitled to a patent unless –

(b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

4. Claims 1, 10, and 12 are rejected under 35 U.S.C. § 102(b) as being anticipated by US Patent No. 5,371,878 to Coker.

Regarding claims 1 and 12, Coker discloses a method of executing code in a microcontroller (“Embedded Computer System or ECS”, see column 1, lines 20-23) [*“[T]his invention uses hardware and software which can ‘shadow’ the execution of a target-ECS in real time operation”* (column 2, lines 34-40)];

Executing a set of timing code to enable lock-step synchronization in a virtual microcontroller (“shadow system”), wherein the timing code is timed to take the same number of clock cycles as the microcontroller uses to execute boot code, [*“A shadow system of this invention executes the same software as the target-ECS from system start-up or reset.”* (column 2, lines 56-58); *“The shadow system and the target-ECS function exactly the same except that the shadow system receives data slightly delayed because of a data buffer between the target-*

Art Unit: 2123

ECS and the shadow system.” (column 3, lines 13-16); “[I]n terms of relative time, the execution state of the shadow system 28 at the time when any given instruction is executed will directly correspond to the execution state of the target-ECS when the same instruction was executed..” (column 8, lines 44-49)];

Wherein the boot code is stored within the microcontroller and at least one portion of the boot code is inaccessible to the virtual microcontroller [Interface means 19 connecting the target-ECS and shadow system is used by Coker to transmit I/O data and does not appear to contain any suggestion that instruction code or boot code is transmitted via interface means. See (column 4, lines 9-18)]; and

Simultaneously halting both the microcontroller and the virtual microcontroller [the target-ECS and the shadow system execute the same code (column 2, lines 56-58) and execute at in lock-step synchronization (column 8, lines 44-49), therefore when the target-ECS halts, the shadow system simultaneously halts. It is inherent that a computer processor in a debugging system halts because there is a distinct conclusion to the debugging process. An infinitely executing debugging process would fail to achieve its explicit goal of validating the software or hardware].

In response, Applicants argue primarily that:

Coker discloses a shadow system executing the same software as the target-ECS from system start-up or reset (see Coker, col. 2, lines 56-58). Coker further discloses that the shadow system and the target-ECS function exactly the same except that the shadow system receives data slightly delayed (see Coker, col. 3, lines 13-16). Accordingly, Coker discloses a system whereby the target-ECS and the shadow system execute the same code and have the same data but slightly delayed. Independent Claim 1 distinguishes over Coker by reciting a limitation whereby the virtual microcontroller executes a set of timing code as the microcontroller executes the boot code, as claimed whereas Coker discloses executing the same code.

The Examiner respectfully traverses this argument as follows.

The term “timing code” is not a standard term known in the art. Therefore the claim must define the scope of this term. Claim 1 states, “the timing code is timed to take the same number of clock cycles as the microcontroller uses to execute the boot code.” The claim fails to provide any further definition of “timing code.”

The claim language does not exclude the virtual microcontroller from executing the same boot code as the microcontroller, but rather requires that the virtual microcontroller executes code “timed to take the same number of clock cycles as the microcontroller uses to execute the boot code.”

The Examiner agrees with Applicants’ statement that Coker discloses a shadow system executing the same software as the target-ECS from system start-up. Therefore, the shadow system [virtual microcontroller] executes code “timed to take the same number of clock cycles as the target-ECS [microcontroller] uses to execute the boot code.”

Applicants’ argument has been fully considered but has been found unpersuasive.

Applicants’ further argue that:

Moreover, the above referenced Office Action asserts that:

“Interface means 19 connecting the target-ECS and shadow system is used by Coker to transmit I/O data and does not appear to contain any suggestion that instruction code or boot code is transmitted via interface means.”

Claim rejection under 35 U.S.C. 102 cannot be based on speculation nor can it be based on obviousness. The fact that a code does not appear to be transmitted does not necessarily make the code inaccessible, as claimed. A code may not be transmitted but nevertheless accessible. As such, there is a vast difference between a code not being transmitted and a code being inaccessible, as claimed.

...Accordingly, Coker fails to either expressly or inherently disclose or suggest each and every element of the recited limitations of independent Claim 1 including the code being inaccessible, as claimed.

The Examiner respectfully traverses this argument as follows.

The rejection under 35 U.S.C. § 102 entered above is in no way based upon speculation or obviousness, but rather solely on the disclosure found in the prior art.

Applicants' attention is drawn to Coker, FIG. 1, which depicts a one way flow of data from the target-ECS to the shadow system, denoted by the arrows 18 and 26. Coker discloses a system wherein I/O data is *transmitted from the target-ECS to the shadow system*. Coker in no way discloses that the shadow system accesses the target-ECS. Additionally, making the boot code on the target-ECS accessible to the shadow system would be redundant because, as Applicants have noted, the shadow system executes the same code as the target-ECS.

Additionally, there is no evidence whatsoever in the prior art reference that the boot code stored on the target-ECS is *accessible* to the shadow system. That is, there is no disclosure anywhere in the reference that the shadow system may probe, interrogate, or somehow request the boot code from the target-ECS. This conclusion is directly supported by the depiction of a one way flow of data from the target-ECS to the shadow system, denoted by the arrows 18 and 26 in FIG. 1.

Therefore Applicants' allegation that the prior art fails to anticipate this limitation is based on Applicants' speculative interpretation of the reference rather than what the reference discloses.

Applicants' arguments have been fully considered but have been found unpersuasive.

Regarding claim 10, the rationale given above for claim 1 is incorporated and additionally Coker discloses resetting the microcontroller and the virtual microcontroller [“*A shadow system of this invention executes the same software as the target-ECS from system start-up or reset.*” (column 2, lines 56-58)].

The claim recites “setting a break at assembly instruction line zero,” and subsequently “simultaneously halting both the microcontroller and the virtual microcontroller by branching to assembly instruction line zero.” The limitation of “assembly instruction line zero” appears to be an arbitrary location for setting the breakpoint.

Coker discloses copying register and memory contents from the microcontroller to corresponding registers and memory in the virtual microcontroller [*“[T]he shadow system receives its input data from the input registers of the target-ECS and stores in the input data in its RAM... Using the address of unique input events in a specially segregated portion of its RAM, the shadow system central processing unit (CPU) can perform numerical operations on the data with its microprocessor and similarly send outputs to specific locations within its RAM rather than to complex output registers.”* (column 2, line 63 – column 3, line 12)].

The limitation of “removing the break at assembly line zero after copying the register contents and copying the memory contents” is regarded as the necessary and inherent steps of practicing Coker’s invention in order to debug the system. Merely halting execution is not sufficient to implement a debugging system, therefore removing a break is necessary to continue the debugging process.

Claim Rejections - 35 USC § 103

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person

Art Unit: 2123

having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negated by the manner in which the invention was made.

The factual inquiries set forth in *Graham v. John Deere Co.*, 383 U.S. 1, 148 USPQ 459 (1966), that are applied for establishing a background for determining obviousness under 35 U.S.C. 103(a) are summarized as follows:

1. Determining the scope and contents of the prior art.
2. Ascertaining the differences between the prior art and the claims at issue.
3. Resolving the level of ordinary skill in the pertinent art.
4. Considering objective evidence present in the application indicating obviousness or nonobviousness.

This application currently names joint inventors. In considering patentability of the claims under 35 U.S.C. 103(a), the examiner presumes that the subject matter of the various claims was commonly owned at the time any inventions covered therein were made absent any evidence to the contrary. Applicant is advised of the obligation under 37 CFR 1.56 to point out the inventor and invention dates of each claim that was not commonly owned at the time a later invention was made in order for the examiner to consider the applicability of 35 U.S.C. 103(c) and potential 35 U.S.C. 102(e), (f) or (g) prior art under 35 U.S.C. 103(a).

5. Claims 1-2, 4-10, 12, and 14-20 are rejected under 35 U.S.C. § 103(a) as being unpatentable over US Patent No. 6,202,044 to Tzori.

Regarding claims 1 and 12, Tzori teaches an emulation system having a microcontroller, effectively the device under test (DUT), operating in lock-step with a virtual microcontroller, effectively a virtual processor ["*The digital-logic simulation process and the hardware pod may operate in an engaged operating mode ... stimulus/response process*" (column 5, lines 14-24)].

Tzori teaches executing code in the microcontroller (DUT) [*“stimulation-control data to be transmitted to the hardware pod 32 for stimulating a digital logic IC inserted into the IC socket 34”* (column 9, lines 33-37)]. The limitation of executing a “boot code” is regarded as an obvious detail of implementation as Tzori teaches executing instructions in general. Additionally, the “boot code” must be stored on the device under test in order for the device under test to execute that code.

Tzori teaches “executing timing code to enable the lock-step synchronization, wherein the timing code is timed to take the same number of clock cycles” is clearly taught by Tzori by virtue of the method performed by the simulation process and hardware pod [*“stimulus-response cycle”*; *“engaged mode”* (column 9, line 10 – column 10, line 39); *“disengaged mode”* (column 10, line 40 – column 11, line 8); *“re-enters the engaged operating mode”*; *“a delay may occur between transmission of response data from the hardware pod 32 to the simulation process 22, indicated by the line 162, and the resumption of engaged mode operation... Such a delay may occur while the simulation process 22 performs some portion of the digital logic simulation that must be completed before the simulation process 22 may re-enter the engaged operating mode”* (column 11, lines 9-33); Figs. 2 and 3]. When disengaging the hardware pod to execute a set of boot code, it would be obvious to a person of ordinary skill in the art that Tzori teaches that the simulation process must execute “timing code [...] timed to take the same number of clock cycles”. That is, Tzori plainly suggests a combination of lock-step synchronization with “timing code” that allows the microcontroller and virtual processor to optionally disengage from each other [*“An advantage of the improved digital logic simulation/emulation system is that it frees the digital-logic simulation process and the hardware pod from operating in lock-step with each*

Art Unit: 2123

other at the stimulation/response cycle of the digital logic circuit." (column 5, line 64 – column 6, line 1)].

Regarding the step of simultaneously halting, this step is well known in the art as a breakpoint for a concurrent process. In this instance, the concurrent processes are executing in parallel on separate devices (virtual microcontroller and microcontroller, or virtual processor and DUT). Tzori's system and method are clearly conducive to this type of breakpoint, achieved by using the control data during the engaged mode (or invoking the engaged mode if necessary) to simultaneously halt both the microcontroller and virtual microcontroller.

Tzori does not expressly state that the digital logic circuit 34 is a microcontroller. It would have been obvious to a person of ordinary skill in the art to use Tzori's system and method for the particular type of device being designed, whether a microcontroller, a processor, an ASIC, or some other form of integrated circuit logic device. The motivation to do so would be found in the teachings of Tzori as cited above as well as from the nature of the problem to be solved. The combination would be formed according to the teachings of Tzori, where the simulation process and digital logic IC taught by Tzori are modified to correspond to the integrated circuit digital logic device preferred by the designer.

In response, Applicants' argue primarily that:

Tzori discloses that during the initialization interval the server process retrieves and transmits the logic configuration data from the logic-configuration library to the hardware pod and loads it onto the configurable-logic ICs (see Tzori, col. 9, lines 20-25). Tzori further discloses that "after completing the initialization interval, the simulation process performs a sequence of simulation cycles" (see Tzori, col. 9, lines 27-29). Tzori, further continues by describing the method and the process after the initialization interval (see Tzori, col. 9, line 31- to col. 11, line 33). Accordingly, reliance of the above referenced Office Action on the method and the process after the initialization interval, described in Tzori, column 9 line 31

Art Unit: 2123

to column 11 line 33, is misplaced and of absolutely no relevance because the recited limitations of the claimed invention are directed to the booting up procedure of an ICE system.

The Examiner respectfully traverses this argument as follows.

Applicants' argument equates "initialization of configurable-logic ICs" in Tzori to "booting a microcontroller" in claim 1. This argument is defective because these two processes are separate and distinct. The rejection is not based upon an alleged equivalence between initializing configurable-logic ICs and booting a microcontroller. The rejection is based upon the method disclosed by Tzori that occurs *after* the initialization process.

Further, Applicants' arguments may suggest that the claimed invention does not explicitly or implicitly "initialize a configurable-logic IC." However, the specification states at page 9, lines 24-27,

In preferred embodiments, a field programmable gate array FPGA (or other programmable logic device) is configured to function as the virtual microcontroller 220. The FPGA and virtual microcontroller 220 will be referred to interchangeably herein.

It is noted that a "virtual microcontroller" is recited in claim 1 and executes a set of timing code. Therefore it appears that Applicants' claimed invention inherently "initializes a programmable logic device," as does Tzori. If Applicants explicitly interpret the claimed invention as not initializing a configurable-logic IC, such as a programmable logic device or FPGA, clarification is respectfully requested.

Applicants' further argue that:

Moreover, the above referenced Office Action asserts that the limitation of executing a "boot code" is regarded as an obvious detail of implementation as Tzori teaches executing instructions in general. The Applicants respectfully disagree. Assuming *arguendo* that executing a "boot code" is obvious detail of implementation, it is still of no relevance what so ever because as described above, the cited portion of Tzori is directed to after the initialization interval.

The Examiner respectfully traverses this argument as follows.

“Boot code” is merely normal code with an intended purpose. The claim specifies nothing further regarding the nature of “boot code.” Tzori teaches executing code in general. Whether or not the user intends for that code to be “boot code” is not a patentable distinction.

Applicants’ attempt to equate “initializing a configurable-logic IC” to “booting a microcontroller” has been addressed above and found unpersuasive.

Applicants’ arguments have been fully considered but have been found unpersuasive.

Regarding claims 2 and 14, Tzori teaches transmitting response data from the hardware pod (microcontroller or DUT) to the simulation process (virtual microcontroller or virtual processor) (column 11, lines 23-33). This step allows for the simulation process to perform “some portion of the digital logic simulation that must be completed before the simulation process may re-enter the engaged operating mode”. It would be obvious to a person of ordinary skill in the art at the time of Applicants’ invention that synchronizing the simulator process and hardware pod is necessary and performed at this step. As synchronization between the two devices means their registers (real or virtual) and memory contents hold the same values, it would be obvious to a person of ordinary skill in the art at the time of Applicants’ invention to copy the register values and memory contents from one device to the corresponding register values of the other, especially in light of Tzori’s explicit teaching of data transfer between the two when re-entering the engaged operating mode.

Regarding claims 4, 5, 15 and 16, these claims are interpreted as meaning that both the microcontroller (DUT) and virtual microcontroller (virtual processor) branch to the beginning of a section of code following a breakpoint (the simultaneously halting step of claim 1). It would have been obvious to a person of ordinary skill in the art at the time of Applicants' invention to begin a second task at the beginning of that second task upon completion of a first task (boot code or otherwise) when using the system taught by Tzori. Branching to different points in code is extremely well known in the art. If Applicant intends the phrase "branches to assembly instruction line 0" to be read as a literal limitation, clarification is respectfully requested, however the specification (page 28, lines 5-8) appear to teach this phrase as an address stop as known in the art.

Regarding claims 6 and 17, official notice is taken that numerous methods of achieving data protection to effectively "hide data" are well known in the art. It would have been obvious to a person of ordinary skill in the art to hide the boot code from the virtual microcontroller to achieve the numerous advantages of data protection, many of which are known in the art. Applicants have not challenged this use of Official Notice and it is therefore considered admitted prior art. Please see MPEP 2144.03.

Regarding claims 7 and 18, these claims recite setting and initiating a breakpoint, as defined above and known in the art. Official notice is taken that breakpoints are well known in the art. It would have been obvious to a person of ordinary skill in the art at the time of Applicants' invention to implement breakpoints as known in the art.

In response, Applicants traverse this use of Official Notice and request evidentiary support for the statement.

Applicants' attention is drawn to the specification of the application, page 3, lines 1-7, describing the prior art in Applicants' words (emphasis added).

During the course of the analysis, various trace information such as time stamps, register values, data memory content, etc. may be logged in the host computer 110 for analysis and debugging by the designer. **Additionally, it is generally the case that various break points can be defined by the designer that cause the program to halt execution at various points in the operation to permit detailed analysis.** Other debugging tools may also be provided to enable the user to debug the operation of the circuit.

The Examiner respectfully submits that if Applicants' use of the term "breakpoints" in the claims are believed to be patentably distinguishable over Applicants' admitted prior art, clarification is required.

Further, Microsoft Computer Dictionary, Fifth Edition, defines:

breakpoint *n.* A location in a program at which execution is halted so that a programmer can examine the program's status, the contents of variables, and so on. A breakpoint is set and used within a debugger and is usually implemented by inserting at that point some kind of jump, call, or trap instruction that transfers control to the debugger. *See also* debug, debugger.

Claims 8 and 19 recite combinations of the limitations found in claims 2-4 and 7; and 13-15 and 18, respectively. As these claims are obvious in view of Tzori, as set forth above, different combinations of these limitations are similarly obvious in view of Tzori.

Art Unit: 2123

Claims 9 and 20 recite removing a breakpoint. Official notice is taken that removing a breakpoint is well known in the art. It would have been obvious to a person of ordinary skill in the art at the time of Applicants' invention to remove a breakpoint if he no longer wanted execution to break at that instruction.

In response, Applicants traverse this use of Official Notice and request evidentiary support for the statement.

Applicants' attention is drawn to the specification of the application, page 3, lines 1-7, describing the prior art in Applicants' words (emphasis added).

During the course of the analysis, various trace information such as time stamps, register values, data memory content, etc. may be logged in the host computer 110 for analysis and debugging by the designer. **Additionally, it is generally the case that various break points can be defined by the designer that cause the program to halt execution at various points in the operation to permit detailed analysis.** Other debugging tools may also be provided to enable the user to debug the operation of the circuit.

The Examiner respectfully submits that if Applicants' use of the term "breakpoints" in the claims are believed to be patentably distinguishable over Applicants' admitted prior art, clarification is required.

Further, Microsoft Computer Dictionary, Fifth Edition, defines:

breakpoint *n.* A location in a program at which execution is halted so that a programmer can examine the program's status, the contents of variables, and so on. A breakpoint is set and used within a debugger and is usually implemented by inserting at that point some kind of jump, call, or trap instruction that transfers control to the debugger. *See also* debug, debugger.

Inasmuch as Applicants may be alleging that it is unknown to a person of ordinary skill in the art to remove a breakpoint after it has been inserted, the Examiner finds this argument wholly

Art Unit: 2123

unpersuasive. The ubiquity of commercial computer software, ostensibly having been debugged using breakpoints, which executes on the end-user's computer without halting for the far-away programmer to examine the behavior of the program, is evidentiary support that it is well known to remove a breakpoint from computer code for at least the reasons of commercial deployment. Any allegation to the contrary is simply unreasonable.

Claim 10 recites a combination of limitations found in claims 1, 8, and 9. As these claims are obvious in view of Tzori, as set forth above, different combinations of these limitations are similarly obvious in view of Tzori.

Claim 21 is rejected under 35 U.S.C. § 103(a) as being unpatentable over Tzori as applied to claim 12 above, and further in view of "Emulation of the Sparcle Microprocessor with the MIT Virtual Wires Emulation System" by Matthew Dahl, Jonathan Babb, Russel Tessier, Silvina Hanono, David Hoki, and Anant Agarwal (Dahl) and further in view of "A Reconfigurable Logic Machine for Fast Event-Driven Simulation" by Jerry Bauer, Michael Bershteyn, Ian Kaplan, and Paul Vyedin (Bauer).

Tzori teaches that the simulation process (virtual processor) is implemented on a Sun workstation (column 6, line 63-65). Tzori does not explicitly teach that the simulation process is implemented on a field programmable gate array (FPGA).

Dahl teaches that it is known in the art to emulate a Sparc microprocessor using an FPGA (abstract).

Art Unit: 2123

Bauer teaches that hardware emulation can increase simulation speed by up to 10,000 times (introduction, paragraphs 1-2).

Therefore it would have been obvious to a person of ordinary skill in the art at the time of Applicants' invention to combine these teachings and arrive at the decision to implement the simulation process of Tzori, originally implemented on a Sun workstation, on an FPGA to realize an enormous increase in simulation speed. Knowledge that this was possible is provided by Dahl, and motivation to combine the references, to increase simulation speed, is provided by Bauer.

Applicants' arguments directed to claims 8, 10, 19, and 21 have been addressed above in the context of claims 7, 9, 18, and 20.

Conclusion

THIS ACTION IS MADE FINAL. Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire **THREE MONTHS** from the mailing date of this action. In the event a first reply is filed within **TWO MONTHS** of the mailing date of this final action and the advisory action is not mailed until after the end of the **THREE-MONTH** shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event,

Art Unit: 2123

however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

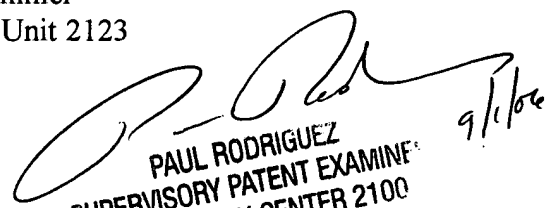
Any inquiry concerning this communication or earlier communications from the examiner should be directed to Jason Proctor whose telephone number is (571) 272-3713. The examiner can normally be reached on 8:30 am-4:30 pm M-F.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Paul Rodriguez can be reached at (571) 272-3753. The fax phone number for the organization where this application or proceeding is assigned is (571) 273-8300.

Any inquiry of a general nature or relating to the status of this application should be directed to the TC 2100 Group receptionist: 571-272-2100. Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see <http://pair-direct.uspto.gov>. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Jason Proctor
Examiner
Art Unit 2123

jsp


PAUL RODRIGUEZ
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100
9/1/04